Detection of Plant Diseases Using Deep Learning
Image Classification
Group 5

1st Sam Knisely
School of Data Science
University of Virginia
Charlottesville VA, USA
sck4jh@virginia.edu

2nd Darreion Bailey
School of Data Science
University of Virginia
Charlottesville VA, USA
rzu5vw@virginia.edu

3rd Alec Pixton
School of Data Science
University of Virginia
Charlottesville VA, USA
etk3pu@virginia.edu

***Abstract*—This paper addresses the critical challenge of enhancing plant disease classification accuracy in real-world agricultural settings, where models trained on idealized, uniform-background datasets often fail. Building upon prior literature suggesting the potential of randomized backgrounds for improved generalization, we augmented the PlantVillage dataset with synthetic farm-color backgrounds and real farm imagery. We utilized lightweight architectures such as MobileNetV2 and EfficientNetB0. These models are less computationally intensive and more suitable for running on a mobile device in the field. The results showed that models trained with synthetic backgrounds had lower accuracies than those trained on original backgrounds. Models trained with farm photo backgrounds had higher test accuracies. The best model identified was an EfficientNetB0 model that yielded a test accuracy of 94.97%.**

I. Motivation

Deep learning models regularly surpass 98 % accuracy on PlantVillage by using CNNs such as VGG-16, ResNet, DenseNet, Xception and even lightweight MobileNet variants [1], [3], [4]. Despite this success, authors highlight two persistent gaps: (i) networks trained on uniform black or gray backgrounds generalize poorly to real farm scenes filled with soil, sky, clutter and occlusion, and (ii) many high-performing architectures are still too heavy for on-device use in low-resource settings [1], [2]. Real-time YOLO-based detectors improve speed but retain limited background diversity and mid-size backbones [2]. Conversely, biodiversity datasets such as LifeCLEF offer rich scenery yet tackle species identification, not disease status [5]. These observations motivate our focus on diversifying backgrounds and benchmarking truly lightweight architectures that can operate on farmers' mobile devices.

We aim to address the challenge of reliably classifying plant diseases in real-world farm conditions, where factors like varying backgrounds, lighting, and occlusions complicate model performance. While many high-accuracy models rely on datasets with uniform or black backgrounds, such approaches often fail to generalize when confronted by the messy realities of agricultural fields. To overcome this limitation, we are experimenting with augmented datasets that incorporate synthetic "farm-like" random backgrounds and real farm imagery. We generated new training images by overlaying leaves onto two types of backgrounds – real farm photos and randomly generated pixel arrays resembling soil, foliage, and sky. Our overarching goal is to build compact, robust models that are accessible to farmers with minimal computing power while handling visually diverse conditions more effectively than uniform-background models.

For our training data, we used the PlantVillage dataset, a widely recognized benchmark for plant disease classification. It consists of 54,000 images across 14 crop species, with labels indicating whether a plant is healthy or diseased, including which disease is present. However, most images in this dataset have uniform black or grayscale backgrounds, which do not reflect real-world farming conditions. The dataset is found at https://github.com/spMohanty/PlantVillage-Dataset

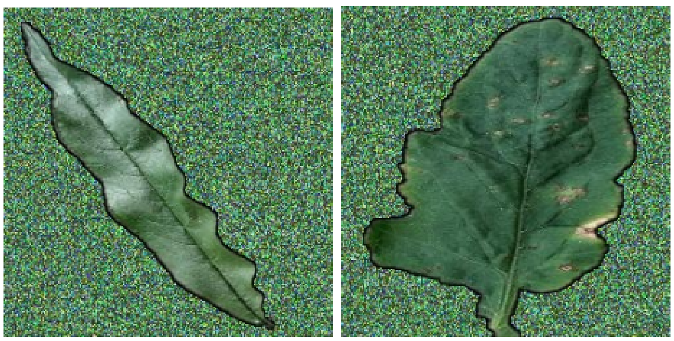
II. Methods

First, we thresholded the mostly black backgrounds in the original PlantVillage images to obtain clean leaf masks. Next, we composited these segmented leaves onto two distinct types of backgrounds: (1) "farm-like" random pixels, where each pixel was sampled from a constrained HSV range of greens, browns, and blues; and (2) real farm photos, which we gathered from open-source libraries. To maintain consistent labeling, we mirrored the original folder structures (e.g., separate subdirectories for each disease type) and appended "_random" or "_farm" to the new image filenames. We also applied small morphological operations (erosion/dilation) to smooth leaf edges and reduce artifacts left by thresholding. Since there are approximately 45,000 original black-background images, running the augmentation pipeline in full produced one new composited version for each of the two background types (random vs. real farm), yielding 90,000 additional images. In total, this led to roughly 135,000 images across all variants (original plus two augmented sets). The process demonstrated how compositing leaves onto various backgrounds can be automated efficiently, marking a vital step toward building more generalizable models. Examples from each dataset are shown in the figures on the next page.

As described earlier, one of our goals is to create a model that farmers can practically use in real-life scenarios. An important part of this is to have a model that is lightweight and is computationally efficient. MobileNetV2 and EfficientNetB0 both meet these criteria and were identified as top lightweight

(a) Healthy (b) Diseased

Fig. 1: Original Black Backgrounds



(a) Healthy (b) Diseased

Fig. 2: Pixel Backgrounds

models in the literature. For each model, we implemented a base case model. Each base case model begins with a frozen base layer followed by a global average pooling layer to reduce overfitting. The pooling layer connects to a dense layer using a ReLU activation function. The final layer is a dense layer using a Sigmoid activation function for binary classification. Finally, we chose the Adam optimizer for faster convergence and stable training. Subsequent models were created by applying additional techniques to these base cases. Our designs incorporated data augmentation, dropout, L2 regularization, and additional dense layers.

(a) Healthy (b) Diseased

Fig. 3: Farm Photo Backgrounds

III. Experiments

To ensure the neural network learned genuine leaf–disease patterns rather than memorizing scenery, we first expanded to the full 54 000-image PlantVillage corpus and generated three new versions of each image: a segmented-leaf variant, a composite on a synthetic "farm-like" pixel mosaic, and a composite on a real farm photograph. We scraped **500** unique farm scenes from open-source repositories and applied color-jitter, lighting shifts, random occlusion masks, and geometric transforms (rotation, scale, saturation). This yielded roughly 162 000 augmented samples, bringing the corpus to 216 000 images. A strict background hold-out rule was enforced: no real or synthetic background appearing in the test fold was used in training or validation. Each disease class was stratified **60 % / 20 % / 20 %** across train, validation, and test sets after the background split, ensuring class balance throughout.

1) Color Jitter: The RGB image is first converted to HSV. During on-the-fly augmentation we multiply the V channel by a uniform factor in $[0.85, 1.15]$ to simulate darker or sunnier scenes, and shift the H channel by a random integer in $[-10°, +10°]$ to emulate sensor-dependent hue drift. Saturation is left unchanged to avoid unrealistic over-saturation.

(a) Random-pixel background (b) Farm-photo background

Fig. 4: Color-jitter augmentation examples.

2) Lighting Shifts (Shadow Mask): With 10 % probability we overlay a semi-transparent black ellipse whose center, axes, and orientation are sampled uniformly within the image bounds. The overlay is blended at $\alpha = 0.4$, producing soft shadows that mimic foliage or equipment casting shade across the leaf.

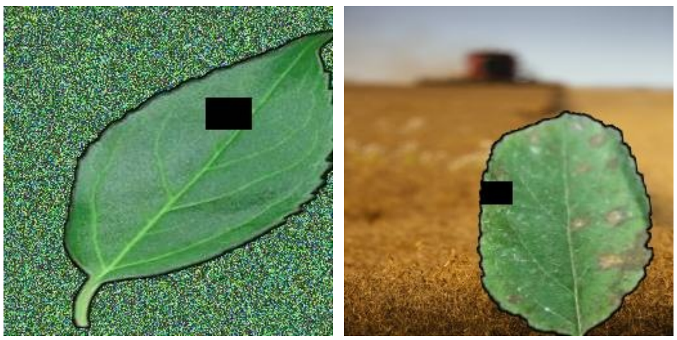
3) Random Occlusion Masks (Cut-out): To force the network to rely on distributed cues rather than a single texture patch, we apply cut-out in 10 % of samples. A rectangle covering 5 – 15 % of the image area is placed at a random location and filled with zeroes, simulating dirt splashes, insect bites, or partial view obstruction.

4) Geometric Transforms: Each image undergoes a compound geometric transform:

- 50 % chance of horizontal flip (mirrors left/right orientation).
- Random rotation drawn from $[-25°, +25°]$.

(a) Random-pixel background (b) Farm-photo background

Fig. 5: Simulated shadow masks for lighting variation.



(a) Random-pixel background (b) Farm-photo background

Fig. 6: Partial occlusion introduced by random cutout.

- Isotropic scaling in the range $[0.7, 1.3]$ to mimic distance variation.
- X/Y translation up to 20 % of image width/height to remove center bias.

The combined affine matrix is applied to both the leaf and its mask using bilinear interpolation (nearest-neighbor for masks).

5) JPEG Compression Artifacts: After compositing, every output is re-encoded at a randomly selected JPEG quality between 60 and 95, then decoded back to BGR. This re-

(a) Random-pixel background (b) Farm-photo background

Fig. 7: Affine geometric transforms (flip, rotate, scale, translate).

(a) Random-pixel background (b) Farm-photo background

Fig. 8: Lossy JPEG-compression artifacts added for realism.

produces blockiness, slight blur, and color banding common in smartphone photos and social-media uploads, making the model robust to real-world compression noise.

As described in the Methods section, we ran a few models with varying specifications. All models were pre-trained on Imagenet, utilized at least one dense layer with ReLU activation functions, had the base layer frozen, used global average pooling, had a final binary classification layer with Sigmoid activation function, and utilized the Adam optimizer.

Four MobileNetv2 models and four EfficientNetB0 models were developed for testing. Three instances of each model were trained, one for each dataset, the segmented dataset, the pixel dataset, and the farm photo dataset. All models were then evaluated on a holdout set from the farm photo dataset to test real-world applicability.

The best performing model based on test accuracy, as described in the Results section, was the EfficientNetB0 model with two dense layers. The confusion matrix for this model is provided below:

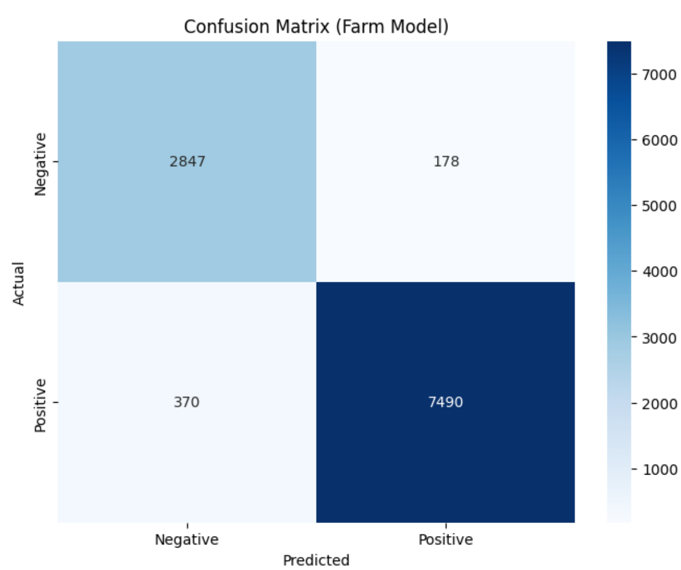


Fig. 9: Confusion Matrix

The confusion matrix shows a false positive rate of 5.84% and a false negative rate of 4.71%. This indicates that the model classifies 5.84% of the negative cases incorrectly as positive and 4.71% of the positive cases incorrectly as negative. The false positive rate is higher, although we may be less concerned with a false positive rather than missing a diseased plant which could potentially allow disease to spread.

IV. Results

The model designs and their test accuracies are listed below:

MobileNetV2 Model 1: This is our base case for MobileNetV2. It includes a frozen base layer, a global average pooling layer, a dense layer, and a binary classification layer.

MobileNetV2 Model 2: This model added to the base case dropout and data augmentation including rotating and zooming the photos.

MobileNetV2 Model 3: This model added to the base case L2 regularization and early stopping.

MobileNetV2 Model 4: This model added to the base case an additional dense layer with a ReLU activation function, L2 regularization, and early stopping.

MobileNetV2 Model Test Accuracies

MobileNetV2	Model 1	Model 2	Model 3	Model 4
Segmented	65.03%	72.50%	72.02%	57.58%
Pixel	74.37%	75.58%	74.47%	75.08%
Farm Photo	83.57%	81.48%	83.58%	83.29%

The table above shows that models 3 and 4 produce the highest test accuracies. These are the models that included L2 regularization and early stopping.

EfficientNetB0 Model 1: This is our base case for EfficientNetB0. It includes a frozen base layer, a global average pooling layer, a dense layer, and a binary classification layer.

EfficientNetB0 Model 2: This model added dropout to the base case.

EfficientNetB0 Model 3: This model added L2 regularization to the base case.

EfficientNetB0 Model 4: This model added an additional dense layer with ReLU activation function to the base case.

EfficientNetB0 Model Test Accuracies

EfficientNetB0	Model 1	Model 2	Model 3	Model 4
Segmented	60.78%	84.53%	81.11%	86.18%
Pixel	74.50%	88.24%	88.96%	88.81%
Farm Photo	94.79%	94.97%	94.63%	94.97%

The table shows only small differences between models for segmented and farm photo models. Pixel models did improve with the addition of techniques, particularly when including an additional dense layer.

For both EfficientNetB0 and MobileNetV2, farm photo models performed the best. All models were tested on holdout sets with real farm backgrounds so it is intuitive that the models trained on farm backgrounds performed the best. Note, the farm backgrounds used for the holdout test set were different than the farm backgrounds used in the training data.

The top performing model from the analysis was the EfficientNetB0 model trained on the farm background photos with two dense layers along with a frozen base layer, global average pooling, and a binary classification layer. This model yielded a test accuracy of 94.97%. Additionally, on the test dataset this model yielded a precision of 97.68%, a recall of 95.29%, and an F1 score of 96.47%.

The datasets used in the analysis along with the scripts for these models and the reported statistics are available at: https://github.com/darreion/Practical-Plant-Disease-Detection-DL/tree/main

V. Conclusions

The confusion matrix ($\mathrm{TN} = 2\,847$, $\mathrm{FP} = 178$, $\mathrm{FN} = 370$, $\mathrm{TP} = 7\,490$) shows a detailed error analysis that extends beyond the theoretical discussion presented earlier. In aggregate, the model attains an overall **accuracy of 94.97 %**, and yields the following additional metrics: sensitivity/recall $= 95.29\,\%$, specificity $= 94.12\,\%$, precision $= 97.68\,\%$, and $F_1 = 96.47\,\%$. These statistics confirm that the lightweight network reliably identifies diseased plants and only occasionally misclassifies healthy/diseased leaves. The false positive/negative rates are 5.84% and 4.71% respectively. This means there is still some discretion required from users of the model. False negatives may be of more consequence due to potentially losing more crops so users should visually confirm negative classifications.

Overall, we found that lightweight model architectures can achieve relatively comparable accuracies (within 4%) to the more computationally expensive models used in many studies. We found EfficientNetB0 models performed better on this task than MobileNetV2 models. Additionally, we found that the training data backgrounds had more impact on model performances than the model designs and architectures. The models using real farm backgrounds saw the greatest accuracies when evaluating on the test data with different farm backgrounds. This reinforces our hypothesis that field-applicable models can be trained using lightweight model architectures with real farm backgrounds applied to the training data. These models could easily be run on a mobile device and produce high accuracies in real-world settings.

A notable takeaway is that more improvement was observed from transforming the training data than altering the model designs. Future researchers may want to continue to explore utilizing real-world applicable backgrounds with the data rather than spending the majority of research on the model specifications.

Overall, the lightweight model architectures using real-world farm backgrounds produced viable, high-performing neural network models that could easily be deployed on mobile devices in the field.

VI. Member Contributions

Darreion was responsible for the data transformations and augmented image generation, which formed the basis for the model evaluations. He developed the pipeline to threshold black backgrounds from the original PlantVillage images, apply morphological operations, and composite leaves onto farm-like random pixel arrays and real farm photos. Additionally, Darreion provided general support for the subsequent model-building and testing processes, and LaTex/Overleaf formatting, helping streamline the overall workflow.

Sam did much of the literature review. He also created the python code to ingest the various datasets, preprocess them for the models, and split the data into training/validation/testing sets. Additionally, he created the MobileNetV2 models and tested various model architectures, as well as some LaTeX/Overleaf formatting.

Alec created models for EfficientNetB0 and tested techniques for improvement. He introduced additional metrics for all models. He also assisted and finalized all writing as well as formatted most text and graphics for LaTeX.

References

[1] Saleem, M. H., Potgieter, J., & Arif, K. M. (2019). Plant Disease Detection and Classification by Deep Learning. Plants, 8(11), 468. https://doi.org/10.3390/plants8110468

[2] Khalid, M., Sarfraz, M. S., Iqbal, U., Aftab, M. U., Niedbała, G., & Rauf, H. T. (2023). Real-Time Plant Health Detection Using Deep Convolutional Neural Networks. Agriculture, 13(2), 510. https://doi.org/10.3390/agriculture13020510

[3] J., A., Eunice, J., Popescu, D. E., Chowdary, M. K., & Hemanth, J. (2022). Deep Learning-Based Leaf Disease Detection in Crops Using Images for Agricultural Applications. Agronomy, 12(10), 2395. https://doi.org/10.3390/agronomy12102395

[4] Saleem, M. H., Potgieter, J., & Arif, K. M. (2020). Plant Disease Classification: A Comparative Evaluation of Convolutional Neural Networks and Deep Learning Optimizers. Plants, 9(10), 1319. https://doi.org/10.3390/plants9101319

[5] Goëau, Hervé & Bonnet, Pierre & Joly, Alexis. (2015). LifeCLEF Plant Identification Task 2015.